**EYE BALL CURSOR MOVEMENT USING OPENCV**

**1 INTRODUCTION**

As the computer technologies are growing rapidly, the importance of human computer interaction becomes highly notable. Some persons who are disabled cannot be able to use the computers. Eye ball movement control mainly used for disabled people. Incorporating this eye controlling system with the computers will make them to work without the help of other individual. Human-Computer Interface (HCI) is focused on use of computer technology to provide interface between the computer and the technology that makes the effective communication between human and computer. Human computer interaction plays the important role .Thus there is a need to find a method that spreads an alternate way for making communication between the human and computer to the individuals those who have impairments and give them an equivalent space to be an element of Information Society [1-5]. In recent years, the human computer interfaces are attracting the attention of various researchers across the globe. Human computer interface is an implementation of the vision-based system for eye movement detection for the disabled people. In the proposed system, we have included the face detection, face tracking, eye detection and interpretation of a sequence of eye blinks in real time for controlling a nonintrusive human computer interface. Conventional method of interaction with the computer with the mouse is replaced with the human eye movements. This technique will help the paralyzed person, physically challenged people especially person without hands to compute efficiently and with the ease of use. Firstly, camera captures the image and focuses on the eye in the image using OpenCV code for pupil detection. This results the center position of the human eye (pupil). Then the center position of the pupil is taken as a reference and based on that the human or the user will control the cursor by moving left and right.

**Objective of the project:**

An individual Human computer interference system is being introduced. In olden times, as an input device the mouse and keyboard were used by human computer interference system. Those people who are suffering from certain disease or illness cannot be able to operate computers. The idea of controlling the computers with the eyes will serve a great use for handicapped and disabled person. Also this type of control will eliminate the help required by other person to handle the computer. This measure will be the most useful for the person who is without hands through which they can operate with the help of their eye movements. The movement of the cursor is directly associated with the center of the pupil. Hence our first step would be detecting the center of point pupil. This process of pupil detection is implemented using the Raspberry Pi and OpenCV. The Raspberry Pi has a SD/MMC card slot which is used for placing the SD card. The SD card is boosted with the operating system that is required starting up of Raspberry Pi. The Raspberry PI will get executed once the application program is loaded into it.

**­­­2. LITERATURE SURVEY**

**Face as mouse through visual face tracking**

This paper introduces a novel camera mouse driven by 3D model based visual face tracking technique. While camera becomes standard configuration for personal computer (PC) and computer speed becomes faster and faster, achieving human machine interaction through visual face tracking becomes a feasible solution to hand-free control. The human facial movement can be decomposed into rigid movement, e.g. rotation and translation, and non-rigid movement, such as the open/close of mouth, eyes, and facial expressions, etc. We introduce our visual face tracking system that can robustly and accurately retrieve these motion parameters from video at real-time. After calibration, the retrieved head orientation and translation can be employed to navigate the mouse cursor, and the detection of mouth movement can be utilized to trigger mouse events. 3 mouse control modes are investigated and compared. Experiments in Windows XP environment verify the convenience of navigation and operations using our face mouse. This technique can be an alternative input device for people with hand and speech disability and for futuristic vision-based game and interface.

**A robust eye gaze tracking method based on a virtual eyeball model**

Gaze positions can provide important cues for natural computer interfaces. In this paper, we describe a new gaze estimation method based on a three dimensional analysis of the human eye which can be used in head-mounted display (HMD) environments. This paper presents four advantages over previous works. First, in order to obtain accurate gaze positions, we used a virtual eyeball model based on the 3D characteristics of the human eyeball. Second, we calculated the 3D position of the virtual eyeball and gaze vector by using a camera and three collimated IR-LEDs. Third, three reference frames (the camera, the monitor and the eye reference frames) were unified, which simplified the complex 3D converting calculations and allowed for calculation of the 3D eye position and gaze position on a HMD monitor. Fourth, a simple user-dependent calibration method was proposed by gazing at one position based on Kappa compensation. Experimental results showed that the eye gaze estimation error of the proposed method was lower than 1°

**A Human-Computer Interface Using Symmetry Between Eyes to Detect Gaze Direction**

In the cases of paralysis so severe that a person's ability to control movement is limited to the muscles around the eyes, eye movements or blinks are the only way for the person to communicate. Interfaces that assist in such communication are often intrusive, require special hardware, or rely on active infrared illumination. A nonintrusive communication interface system called EyeKeys was therefore developed, which runs on a consumer-grade computer with video input from an inexpensive Universal Serial Bus camera and works without special lighting. The system detects and tracks the person's face using multiscale template correlation. The symmetry between left and right eyes is exploited to detect if the person is looking at the camera or to the left or right side. The detected eye direction can then be used to control applications such as spelling programs or games. The game ldquoBlockEscaperdquo was developed to evaluate the performance of EyeKeys and compare it to a mouse substitution interface. Experiments with EyeKeys have shown that it is an easily used computer input and control device for able-bodied people and has the potential to become a practical tool for people with severe paralysis.

**Facial Feature Based Method For Real Time Face Detection and Tracking I-CURSOR**

This project aims to present an application that is able of replacing the traditional mouse with the human face as a new way to interact with the computer. Facial features (nose tip and eyes) are detected and tracked in real-time to use their actions as mouse events. In our work we were trying to compensate people who have hands disabilities that prevent them from using the mouse by designing an application that uses facial features (nose tip and eyes) to interact with the computer. It can be applied to a wide range of face scales. Our basic strategy for detection is fast extraction of face candidates with a Six-Segmented Rectangular (SSR) filter and face verification by a support vector machine. A motion cue is used in a simple way to avoid picking up false candidates in the background. In face tracking, the patterns of between-the eyes are tracked with updating template matching.

**Real-time eye-gaze estimation using a low-resolution webcam**

Eye detection and gaze estimation play an important role in many applications, e.g., the eye-controlled mouse in the assisting system for disabled or elderly persons, eye fixation and saccade in psychological analysis, or iris recognition in the security system. Traditional research usually achieves eye tracking by employing intrusive infrared-based techniques or expensive eye trackers. Nowadays, there are more and more needs to analyze user behaviors from tracking eye attention in general applications, in which users usually use a consumer-grade computer or even laptop with an inexpensive webcam. To satisfy the requirements of rapid developments of such applications and reduce the cost, it is no more practical to apply intrusive techniques or use expensive/specific equipment. In this paper, we propose a real-time eye-gaze estimation system by using a general low-resolution webcam, which can estimate eye-gaze accurately without expensive or specific equipment, and also without an intrusive detection process. An illuminance filtering approach is designed to remove the influence from light changes so that the eyes can be detected correctly from the low-resolution webcam video frames. A hybrid model combining the position criterion and an angle-based eye detection strategy are also derived to locate the eyes accurately and efficiently. In the eye-gaze estimation stage, we employ the Fourier Descriptor to describe the appearance-based features of eyes compactly. The determination of eye-gaze position is then carried out by the Support Vector Machine. The proposed algorithms have high performances with low computational complexity. The experiment results also show the feasibility of the proposed methodology

**Using Kernels for avideobased mouse-replacement interface**

Some people cannot use their hands to control a computer mouse due to conditions such as cerebral palsy or multiple sclerosis. For these individuals, there are various mouse-replacement solutions. One approach is to enable them to control the mouse pointer using head motions captured with a web camera. One such system, the Camera Mouse, uses an optical flow approach to track a manually-selected small patch of the subject’s face, such as the nostril or the edge of the eyebrow. The optical flow tracker may lose the facial feature when the tracked image patch drifts away from the initially-selected feature or when a user makes a rapid head movement. To address the problem of feature loss, we developed and incorporated the Kernel-Subset-Tracker into the Camera Mouse. The Kernel-Subset-Tracker is an exemplar-based method that uses a training set of representative images to produce online templates for positional tracking. We designed the augmented Camera Mouse so that it can compute these templates in real time, employing kernel techniques traditionally used for classification. We propose three versions of the Kernel-Subset-Tracker, each using a different kernel, and compared their performance to the optical-flow tracker under five different experimental conditions. Our experiments with test subjects show that augmenting the Camera Mouse with the Kernel-Subset-Tracker improves communication bandwidth statistically significantly. Tracking of facial features was accurate, without feature drift, even during rapid head movements and extreme head orientations. We conclude by describing how the Camera Mouse augmented with the Kernel-Subset-Tracker enabled a stroke-victim with severe motion impairments to communicate via an on-screen keyboard.

**Hardware and software implementation of real time electrooculogram (EOG) acquisition system to control computer cursor with eyeball movement**

Human computer interface (HCI) is an emerging technology of neuroscience and artificial intelligence. Development of HCI system using bio signal e.g. Electrooculogram (EOG), Electromyogram (EMG), Electroencephalogram (EEG), Functional near-infrared spectroscopy (fNIRS) etc. are attracted more and more attention of researchers all over the world in recent years because through this it is possible to get acquainted with advanced technologies of artificial intelligence. This paper presents the design and implementation of a fully functional Electrooculogram (EOG) based human computer interface. In this work we have designed and implemented necessary hardware and software for EOG signal acquisition along with controlling hardware such as wheelchair, robotic arm, mobile robot etc., and move computer mouse cursor simultaneously using EOG signal. This interface has three portion: EOG signal acquisition and amplification, analog to digital conversion, and real time hardware and mouse cursor movement. Eye movement is detected by measuring potential difference between cornea and retina using five Ag-Agcl disposable electrodes. Frequency range of EOG signal is considered as 0.3 to 15Hz, so this frequency range is taken using an active high and low pass filter so that accurate EOG signal can be achieved. The analog output of the EOG signal from filter is converted into digital signal by using an Arduino. Arduino serialize the EOG data for calibration and provides a threshold reference point which is used for controlling Hardware. The Classification module e.g. Support Vector machine (SVM) and Linear Discriminant Analysis (LDA) classify live data with respect to the horizontal and vertical data. This works as a binary classifier and choose optimal hyper-plane between two variables. According to each update on the eye position, cursor automatically accelerated in particular direction. PyMouse module in python is used for this task. Eye gesture based Hardware like robot, wheelchai...

**Method for controlling device on the basis of eyeball motion, and device therefor**

A method of controlling an operation of a display device using eye movements and a device for performing the method are provided. The method includes receiving eye movement information of a user; receiving blinking information of the user; generating a control command corresponding to the eye movement information and the blinking information; and controlling an operation of the display device based on the generated control command.

**An SSVEPactuated brain computer interface using phase-tagged flickering sequences: a cursor system**

This study presents a new steady-state visual evoked potential (SSVEP)-based brain computer interface (BCI). SSVEPs, induced by phase-tagged flashes in eight light emitting diodes (LEDs), were used to control four cursor movements (up, right, down, and left) and four button functions (on, off, right-, and left-clicks) on a screen menu. EEG signals were measured by one EEG electrode placed at Oz position, referring to the international EEG 10-20 system. Since SSVEPs are time-locked and phase-locked to the onsets of SSVEP flashes, EEG signals were bandpass-filtered and segmented into epochs, and then averaged across a number of epochs to sharpen the recorded SSVEPs. Phase lags between the measured SSVEPs and a reference SSVEP were measured, and targets were recognized based on these phase lags. The current design used eight LEDs to flicker at 31.25 Hz with 45 degrees phase margin between any two adjacent SSVEP flickers. The SSVEP responses were filtered within 29.25-33.25 Hz and then averaged over 60 epochs. Owing to the utilization of high-frequency flickers, the induced SSVEPs were away from low-frequency noises, 60 Hz electricity noise, and eye movement artifacts. As a consequence, we achieved a simple architecture that did not require eye movement monitoring or other artifact detection and removal. The high-frequency design also achieved a flicker fusion effect for better visualization. Seven subjects were recruited in this study to sequentially input a command sequence, consisting of a sequence of eight cursor functions, repeated three times. The accuracy and information transfer rate (mean +/- SD) over the seven subjects were 93.14 +/- 5.73% and 28.29 +/- 12.19 bits/min, respectively. The proposed system can provide a reliable channel for severely disabled patients to communicate with external environments.

**Deep Learning-Based Real-Time AI Virtual Mouse System Using Computer Vision to Avoid COVID-19 Spread**

The mouse is one of the wonderful inventions of Human-Computer Interaction (HCI) technology. Currently, wireless mouse or a Bluetooth mouse still uses devices and is not free of devices completely since it uses a battery for power and a dongle to connect it to the PC. In the proposed AI virtual mouse system, this limitation can be overcome by employing webcam or a built-in camera for capturing of hand gestures and hand tip detection using computer vision. The algorithm used in the system makes use of the machine learning algorithm. Based on the hand gestures, the computer can be controlled virtually and can perform left click, right click, scrolling functions, and computer cursor function without the use of the physical mouse. The algorithm is based on deep learning for detecting the hands. Hence, the proposed system will avoid COVID-19 spread by eliminating the human intervention and dependency of devices to control the computer.

**FACE CURSOR MOVEMENT USING OPEN CV**

Some peoples cannot able to operate computers because of an illness. The idea of eye controls of great use to not only the future of natural input but more importantly the handicapped and disabled. Moreover, implementing a controlling system in it enables them to operate computer without the help of another person. It is more helpful to handicapped peoples. Those are need to operate computers without hand this one is most useful those can operate cursor by movement of eye. In this paper Camera is capturing the image of eye movement. First detect pupil center position of eye. Then the different variation on pupil position gets different movement of cursor. What all these applications have in common is that the use of personal computers is mostly based on the input method via keyboard and mouse. While this is not a problem for a healthy individual, this may be an insurmountable bound for people with limited freedom of movement of their limbs. In these cases it would be preferable to use input methods which are based on more abilities of the region such as eye movements. To enable such substitute input methods a system was made which follows a low-price approach to control a mouse cursor on a computer system. The eye tracker is based on images recorded by a mutated webcam to acquire the eye movements. These eye movements are then graphed to a computer screen to position a mouse cursor accordingly. The movement of mouse by automatically adjusting the position where of eyesight. Camera is used to capture the image of eye movement. Controlling of mouse cursor is obtained by face movement as moving face up, down, left and right and mouse events are controlled through eye blinks and voice. To perform these operations different algorithms like Haar Cascade algorithm, Template Matching and Hough transformation are used. Our system is mainly aimed for disabled peoples to have effective communication with computer. There are different reasons for which people need an artificial of locomotion such as a virtual keyboard. The number of people, who need to move around with the help of some article means, because of an illness. Moreover, implementing a controlling system in it enables them to move without the help of another person is very helpful. The idea of face controls is of great use to not only the future of natural input but more importantly the handicapped and disabled. It uses various image processing methods such as face detection, eye extraction. It uses a typical webcam to capture an input image. Camera is capturing the image of eyes, mouth, nose and head movement. First detect pupil center position of nose. Then the different variation on face position get different command set for virtual keyboard. The signals pass the motor driver to interface with the virtual keyboard itself. The motor driver will control both speed and direction to enable the virtual keyboard to move forward, left, right and stop.

**Face as mouse through visual face tracking**

This paper introduces a novel camera mouse driven by visual face tracking based on a 3D model. As the camera becomes standard configuration for personal computers (PCs) and computation speed increases, achieving human-machine interaction through visual face tracking becomes a feasible solution to hands-free control. Human facial movements can be broken down into rigid motions, such as rotation and translation, and non-rigid motions such as opening, closing, and stretching of the mouth. First, we describe our face tracking system which can robustly and accurately retrieve these motion parameters from videos in real time [H. Tao, T. Huang, Explanation-based facial motion tracking using a piecewise Bezier volume deformation model, in: Proceedings of IEEE Computer Vision and Pattern Recogintion, vol. 1, 1999, pp. 611-617]. The retrieved (rigid) motion parameters can be employed to navigate the mouse cursor; the detection of mouth (non-rigid) motions triggers mouse events in the operating system. Three mouse control modes are investigated and their usability is compared. Experiments in the Windows XP environment verify the convenience of our camera mouse in hands-free control. This technology can be an alternative input option for people with hand and speech disability, as well as for futuristic vision-based games and interfaces.

[**Cursor Control through Eye movement**](https://www.ijcseonline.org/pdf_paper_view.php?paper_id=5445&9-IJCSE-08770-26.pdf)

Eye tracking system has a suitable design which controls any devices which has digital screen with the eyeball movement and gesture without any help of required hardware. This technical concept has a potential to abolish and replace the standard mouse with the human eyes as a new way to interact and communicate with computer and also intended to replace standard computer screen pointing devices for the use of disable and handicapped people or as an alternative for using mouse which is very easy to use for faster input process. The system makes use of a PC webcam in order to detect eye movement. The system continuously scans camera input for pattern similar to the eye. Once the eye is detected, the system locks it as an object. The eye moment image is captured and transmitted by Raspberry Pi 3 model b and Microcontroller in order to process with OpenCV to derive the coordinator of the eyeball. The approach we described and defined is a real-time, non-intrusive, quick and cost-effective method of tracking facial features with the help of IR sensors.

**3 .SYSTEM ANALYSIS**

**3.1 Existing System**

Gaze estimation can be used in Head-mounted display (HMD) environments since they can afford important natural computer interface cues. This new gaze estimation is based on 3D analysis of human eye. There are various commercial products which use gaze detection technology. In this method, the user has to point only one point for calibration it will then estimate the gaze points. The facial features such as eyes and nose tip are recognized and tracked to avoid the traditional mouse movements with the human face for human interaction with the computer. This method can be applied to face scales in a wide range.

**Disadvantages**

1.Less accuracy

**3.2 PROPOSED SYSTEM**

The proposed system uses Raspberry Pi board of version 3, which is attached with the Monitor, PIR Sensor, and Camera. These materials are attached by USB adaptors. Raspberry plays a vital role in the working module that keeps the eye movement with sensors. Raspberry pi uses SD card, to install raspbian is along with programming codes. PIR sensor also used for detecting human movement.

**Advantages**

1.High accuracy

**Modules:**

**Video Recording:** Using this module we will connect application to webcam using OPENCV built-in function called VideoCapture.

**Frame Extraction**: Using this module we will grab frames from webcam and then extract each picture frame by frame and send that frame to GazeTracking.

**GazeTracking:** Using this module we can detect eyeballs and the extract x and y coordinates of both left and right pupil.

**MoveCursor:** Using this module we will instruct mouse to change its current location to given new x and y location.

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

  
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artefacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labour data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behaviour of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Non-functional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms *what* must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.4.2. External Interface Requirements**

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**Operating Environment**

Windows XP.

**HARDWARE REQUIREMENTS:**

# Processor - Pentium –IV

* Speed - 1.1 Ghz
* RAM - 256 MB(min)
* Hard Disk - 20 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

* Operating System - Windows7/8
* Programming Language - Python

**4. SYSTEM DESIGN**

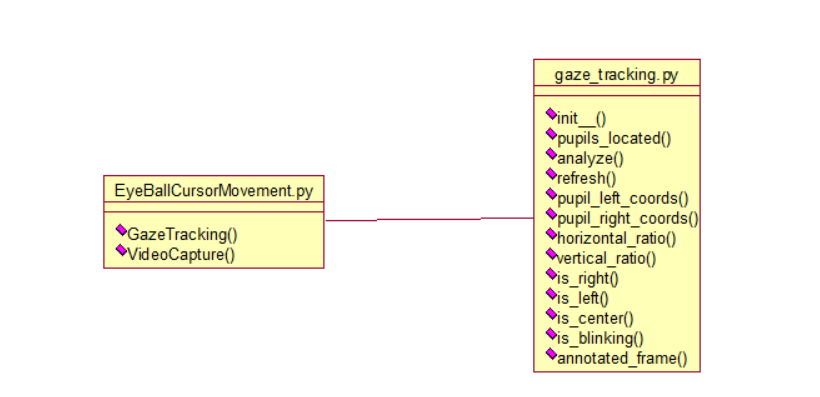
**UML Diagram:**

**Class Diagram:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake

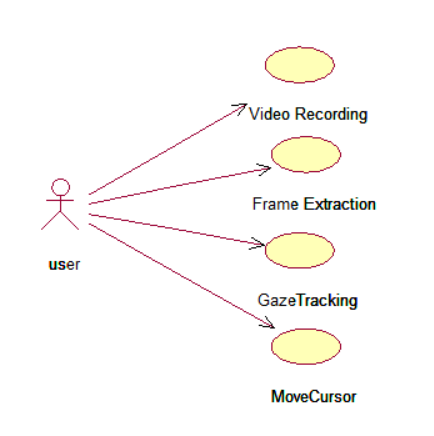
**Class Diagram:**



**Use case Diagram:**

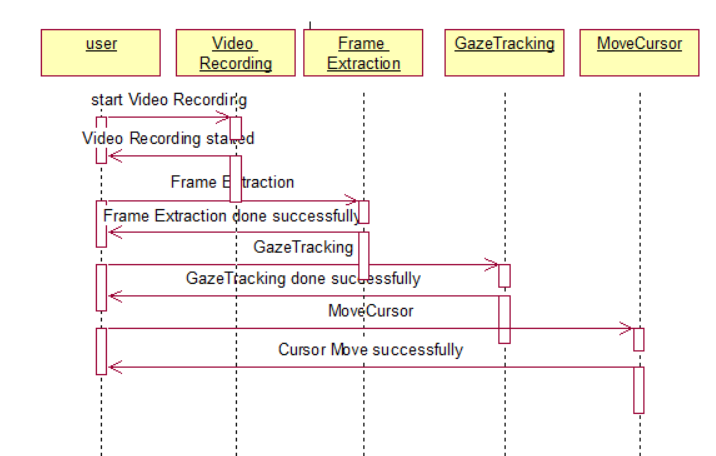
A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

**Use case Diagram:**



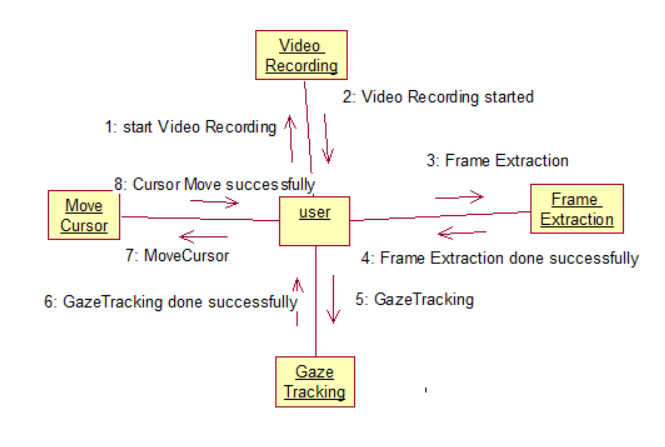
**Sequence diagram:**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**Collaboration diagram:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behavior of a system.

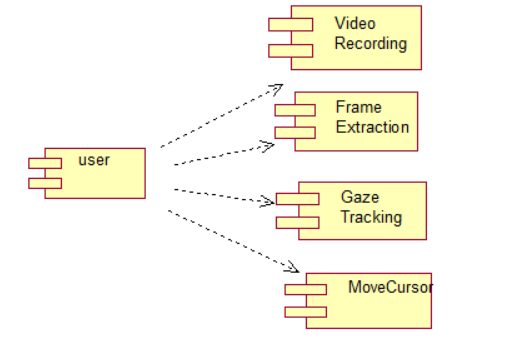


**Component Diagram:**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.

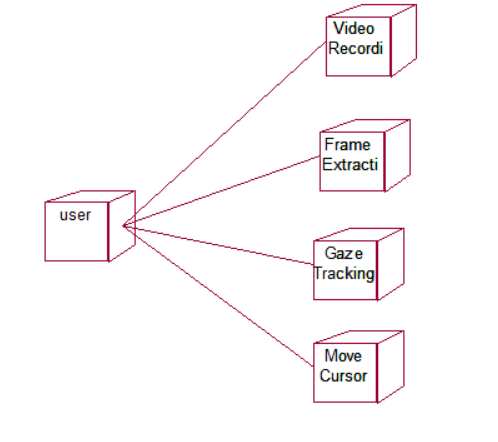
**Component Diagram:**



**Deployment Diagram:**

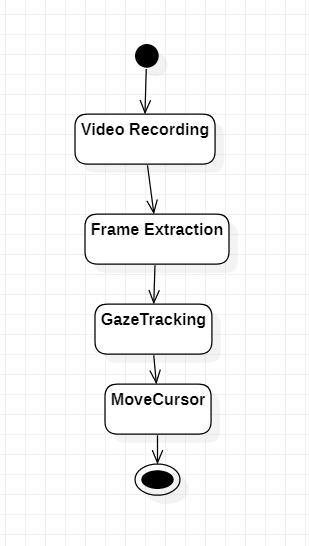
A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.



**Activity Diagram:**

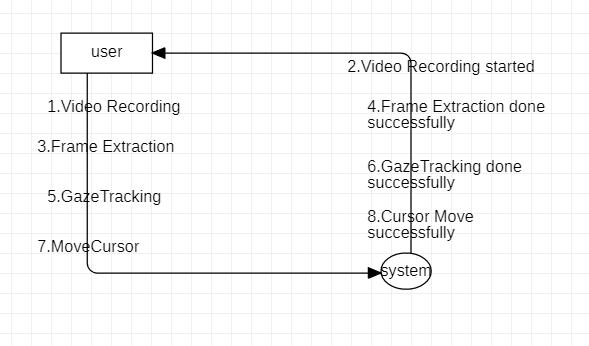
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent



**Data Flow Diagram:**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

**EyeBallCursorMovement.py**

"""

moving mouse cursor using opencv eyeball tracking logic

to move cursor we are using pyautogui

"""

#python library import statement

import cv2

from gaze\_tracking import GazeTracking #python opencv gaze library to track eye ball movement

import pyautogui

gaze = GazeTracking() #eye ball tracking object creation

webcam = cv2.VideoCapture(0) #starting web cam

while True:

\_, frame = webcam.read() #reading frames from webcam

gaze.refresh(frame)#sending frame to opencv gaze library to detect eye ball movement

frame = gaze.annotated\_frame() #returns eye ball movement data

text = ""

if gaze.is\_blinking(): #displaying result

text = "Blinking"

elif gaze.is\_right():

text = "Looking right"

elif gaze.is\_left():

text = "Looking left"

elif gaze.is\_center():

text = "Looking center"

cv2.putText(frame, text, (90, 60), cv2.FONT\_HERSHEY\_DUPLEX, 1.6, (147, 58, 31), 2)

left\_pupil = gaze.pupil\_left\_coords()

right\_pupil = gaze.pupil\_right\_coords() #getting pupil location as x and y cordinates

x = str(left\_pupil).split(",") #getting left pupil x and y location

y = str(right\_pupil).split(",") #getting right pupil x and y location

if len(x) > 1:

data\_x = x[0]

data\_x = data\_x[1:len(data\_x)];

data\_y = x[1]

data\_y = data\_y[0:len(data\_y)-1]

pyautogui.moveTo(int(data\_x),int(data\_y)) #moving mouse cursor to eye pupil x and y left side location

if len(y) > 1:

data\_x = y[0]

data\_x = data\_x[1:len(data\_x)];

data\_y = y[1]

data\_y = data\_y[0:len(data\_y)-1]

pyautogui.moveTo(int(data\_x),int(data\_y)) #moving mouse cursor to eye pupil x and y right sidelocation

cv2.putText(frame, "Left pupil: " + str(left\_pupil), (90, 130), cv2.FONT\_HERSHEY\_DUPLEX, 0.9, (147, 58, 31), 1)

cv2.putText(frame, "Right pupil: " + str(right\_pupil), (90, 165), cv2.FONT\_HERSHEY\_DUPLEX, 0.9, (147, 58, 31), 1)

cv2.imshow("EyeBall Cursor Movement", frame)

if cv2.waitKey(1) == 27:

break

**gaze\_tracking.py**

from \_\_future\_\_ import division

import os

import cv2

import dlib

from .eye import Eye

from .calibration import Calibration

class GazeTracking(object):

"""

This class tracks the user's gaze.

It provides useful information like the position of the eyes

and pupils and allows to know if the eyes are open or closed

"""

def \_\_init\_\_(self):

self.frame = None

self.eye\_left = None

self.eye\_right = None

self.calibration = Calibration()

# \_face\_detector is used to detect faces

self.\_face\_detector = dlib.get\_frontal\_face\_detector()

# \_predictor is used to get facial landmarks of a given face

cwd = os.path.abspath(os.path.dirname(\_\_file\_\_))

model\_path = os.path.abspath(os.path.join(cwd, "trained\_models/shape\_predictor\_68\_face\_landmarks.dat"))

self.\_predictor = dlib.shape\_predictor(model\_path)

@property

def pupils\_located(self):

"""Check that the pupils have been located"""

try:

int(self.eye\_left.pupil.x)

int(self.eye\_left.pupil.y)

int(self.eye\_right.pupil.x)

int(self.eye\_right.pupil.y)

return True

except Exception:

return False

def \_analyze(self):

"""Detects the face and initialize Eye objects"""

frame = cv2.cvtColor(self.frame, cv2.COLOR\_BGR2GRAY)

faces = self.\_face\_detector(frame)

try:

landmarks = self.\_predictor(frame, faces[0])

self.eye\_left = Eye(frame, landmarks, 0, self.calibration)

self.eye\_right = Eye(frame, landmarks, 1, self.calibration)

except IndexError:

self.eye\_left = None

self.eye\_right = None

def refresh(self, frame):

"""Refreshes the frame and analyzes it.

Arguments:

frame (numpy.ndarray): The frame to analyze

"""

self.frame = frame

self.\_analyze()

def pupil\_left\_coords(self):

"""Returns the coordinates of the left pupil"""

if self.pupils\_located:

x = self.eye\_left.origin[0] + self.eye\_left.pupil.x

y = self.eye\_left.origin[1] + self.eye\_left.pupil.y

return (x, y)

def pupil\_right\_coords(self):

"""Returns the coordinates of the right pupil"""

if self.pupils\_located:

x = self.eye\_right.origin[0] + self.eye\_right.pupil.x

y = self.eye\_right.origin[1] + self.eye\_right.pupil.y

return (x, y)

def horizontal\_ratio(self):

"""Returns a number between 0.0 and 1.0 that indicates the

horizontal direction of the gaze. The extreme right is 0.0,

the center is 0.5 and the extreme left is 1.0

"""

if self.pupils\_located:

pupil\_left = self.eye\_left.pupil.x / (self.eye\_left.center[0] \* 2 - 10)

pupil\_right = self.eye\_right.pupil.x / (self.eye\_right.center[0] \* 2 - 10)

return (pupil\_left + pupil\_right) / 2

def vertical\_ratio(self):

"""Returns a number between 0.0 and 1.0 that indicates the

vertical direction of the gaze. The extreme top is 0.0,

the center is 0.5 and the extreme bottom is 1.0

"""

if self.pupils\_located:

pupil\_left = self.eye\_left.pupil.y / (self.eye\_left.center[1] \* 2 - 10)

pupil\_right = self.eye\_right.pupil.y / (self.eye\_right.center[1] \* 2 - 10)

return (pupil\_left + pupil\_right) / 2

def is\_right(self):

"""Returns true if the user is looking to the right"""

if self.pupils\_located:

return self.horizontal\_ratio() <= 0.35

def is\_left(self):

"""Returns true if the user is looking to the left"""

if self.pupils\_located:

return self.horizontal\_ratio() >= 0.65

def is\_center(self):

"""Returns true if the user is looking to the center"""

if self.pupils\_located:

return self.is\_right() is not True and self.is\_left() is not True

def is\_blinking(self):

"""Returns true if the user closes his eyes"""

if self.pupils\_located:

blinking\_ratio = (self.eye\_left.blinking + self.eye\_right.blinking) / 2

return blinking\_ratio > 3.8

def annotated\_frame(self):

"""Returns the main frame with pupils highlighted"""

frame = self.frame.copy()

if self.pupils\_located:

color = (0, 255, 0)

x\_left, y\_left = self.pupil\_left\_coords()

x\_right, y\_right = self.pupil\_right\_coords()

cv2.line(frame, (x\_left - 5, y\_left), (x\_left + 5, y\_left), color)

cv2.line(frame, (x\_left, y\_left - 5), (x\_left, y\_left + 5), color)

cv2.line(frame, (x\_right - 5, y\_right), (x\_right + 5, y\_right), color)

cv2.line(frame, (x\_right, y\_right - 5), (x\_right, y\_right + 5), color)

return frame

**6. TESTING**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

**Implementation**

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

**Testing**

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

**System Testing**

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

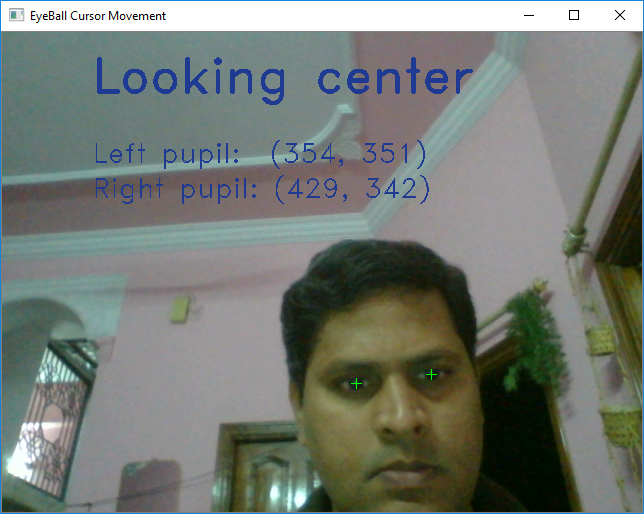
**Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | **Actual** |
| 01 | Video Recording | Test whether the Video is Recording or not. | If Video is not Recording | we cannot do further operations | If Video is Recording we will do further operations | High | High |
| 02 | Frame Extraction | Test whether grab frames from webcam and then extract each picture  or not | Without grabbing and extracting | We cannot do further operations | We Can do further operations | High | High |
| 03 | GazeTracking | Test whether the detecting eyeballs and the extract x and y coordinates or not | Without detecting eyeballs | we cannot do further operations | we Can do further operations | High | High |
| 04 | **MoveCursor** | Verify the cursor is changing its current location or not | Without changing its current location | we cannot detect eye ball movements | we can detect eye ball movements | High | High |

**7. SCREENSHOTS:**

To run this project double click on ‘run.bat’ file to get below webcam screen.



In above screen you can see cursor moves based on eye ball movement. Exception will raise and window close if u move cursor close corners of the screen

**8. CONCLUSION**

From the process implemented it is cleared that the cursor can be controlled by the eyeball movement i.e., without using hands on the computers. This will be helpful for the people having disability in using the physical parts of the computers to control the cursor points. Because the cursor points can be operated by moving the eyeballs. Without the help of others disabled people can use the computers. This technology can be enhanced in the future by inventing more techniques like clicking events as well as to do all the mouse movements and also for human interface systems using eye blinks. Technology also extended to the eyeball movement and eye blinking to get the efficient and accurate movement.

**9.REFERENCES**

[1] Jilin Tu, Thomas Huang, Elect and Comp EngrDept, Hai Tao, ElectEnggDept, “Face as Mouse through Visual Face Tracking”,IEEE,(2005).

[2] EniChul Lee Kang Ryoung Park “A robust eye gaze tracking methodbased on a virtual eyeball model”, Springer, pp.319-337, Apr (2008).

[3] John J. Magee, MargritBetke, James Gips, Matthew R. Scott, and Benjamin N.Waber“A Human-Computer Interface Using Symmetry Between Eyes to Detect Gaze Direction” IEEE Trans, Vol. 38, no.6,pp.1248-1259, Nov(2008).

[4] SunitaBarve, DhavalDholakiya, Shashank Gupta, DhananjayDhatrak, “Facial Feature Based Method For Real Time Face Detection and Tracking I-CURSOR”, International Journal of EnggResearchand App., Vol. 2, pp. 1406-1410, Apr (2012).

[5] Yu-Tzu Lin Ruei-Yan Lin Yu-Chih Lin Greg C Lee “Real-time eye-gaze estimation using a low-resolution webcam”, Springer, pp.543-568, Aug (2012).

[6] Samuel Epstein-Eric MissimerMargritBetke “Using Kernels for avideobased mouse-replacement interface”, Springer link, Nov (2012)

[7] Hossain, Zakir, Md Maruf Hossain Shuvo, and Prionjit Sarker. "Hardware and software implementation of real time electrooculogram (EOG) acquisition system to control computer cursor with eyeball movement." In 2017 4th International Conference on Advances in Electrical Engineering (ICAEE), pp. 132-137. IEEE, 2017.

[8] Lee, Jun-Seok, Kyung-hwa Yu, Sang-won Leigh, Jin-Yong Chung, and Sung-Goo Cho. "Method for controlling device on the basis of eyeball motion, and device therefor." U.S. Patent 9,864,429, issued January 9, 2018.

[9] Lee, Po-Lei, Jyun-Jie Sie, Yu-Ju Liu, Chi-Hsun Wu, Ming-Huan Lee, Chih- Hung Shu, Po-Hung Li, Chia-Wei Sun, and Kuo-Kai Shyu. "An SSVEPactuated brain computer interface using phase-tagged flickering sequences: a cursor system." Annals of biomedical engineering 38, no. 7 (2010): 2383-2397